Machine Translation in India

DR.RAJENDRNAN, S

AMRITA VISHWA VIDYAPEETHAM

COIMBATORE

rajushush@gmail.com

Machine Translation activities in India

- Machine Translation activities in India are relatively young. The earliest efforts date from the mid 80s and early 90s.
- The prominent among these efforts are the research and development projects at
- Indian Institute of Technology (IIT) Kanpur (http://www.cse.iitk.ac.in/users/rmk/proj/proj.html#mt),
- Computer and Information Sciences Department, University of Hyderabad (http://www.uohyd.ernet.in/),
- National Center for Software Technology (NCST), Mumbai
- (now, Center for Development of Advanced Computing (CDAC), Mumbai) (http://www.ncst.ernet.in/) and
- Center for Development of Advanced Computing (CDAC), Pune (http://www.cdac.in/).

Machine Translation activities in India

- The Department of Information Technology, Ministry of Communications and Information Technology, Government of India, through its Technology Development in Indian Languages (TDIL) Project (http://tdil.mit.gov.in) since 1990-‘91 and
- Department of Official Languages, Ministry of Home Affairs, Government of India, since 1989-90 have played instrumental roles by funding these projects.

Machine Aided Translation in India

- The department of Electronics, Govt. of India has started a project called Technology Development for Indian Languages (TDIL) in 1991 and supporting a number of research institutes in the country for the development of all the 18 scheduled languages.

Machine Aided Translation in India

- Most of these systems are in the English-Hindi or Indian language-Indian language domain with exceptions of a Hindi-English and an English-Kannada machine translation system.
- The translation domains are mostly government documents/reports and news stories.
- There are a number of other MT systems that are at their various phases of development and have been demonstrated at various forums.
- Many of these systems cover other Indian languages beside Hindi.

Machine Aided Translation in India

- The following machine translation systems are either being used in field-testing or are available as a web service.
- Most of these systems are in the English-Hindi language domain with exceptions of a Hindi-English and an English-Kannada machine translation system.

Machine Translation in India

- MT Research in lndia was started in early 90's. Most of the operational and experimental systems are rule-based.
- Some important projects include:
  - Mantra by CDAC, Pune
  - Anusaaraka by IlT-Kanpur and University of Hyderabad
  - AnglaBharti by IlT Kanpur
  - Shakti by LTRC, IlIT Hyderabad
  - MaTra by CDAC, Mumbai
- IL-IL MT Consortia project: Language pairs include Marathi - Hindi, Bengali- Hindi, Telugu - Hindi, Hindi- Tamil-Telugu, Malayalam-Tamil
- E-IL MT Consortia project: Language pairs include English Hindi, Marathi, Oriya, Urdu and Bengali
-

ANGLABHARTI and ANUBHARTI Technology

- The ANGLABHARTI project was launched by Professor R. M. K. Sinha at the Indian Institute of Technology, Kanpur in 1991 for machine aided translation from English to Indian languages.
- Professor Sinha has pioneered MT research in India. The system's approach and lexicon is general-purpose with provision for domain customization.
- The system has been applied in several domains such as public health compaign, routine office-correspondance, technical-mannual etc.

ANGLABHARTI

- The first prototype was built for English to Tamil in 1991 and later a more comprehensive system was built for English to Hindi translation.
- AnglaBharti is a pattern directed rule based system with context free grammar like structure for English (source language).
- It generates a ‘pseudo-target’ (Pseudo-Interlingua) applicable to a group of Indian languages (target languages) such as Indo-Aryan family (Hindi, Bengali, Assamese, Punjabi, Marathi, Oriya, Gujarati etc.), Dravidian family (Tamil, Telugu, Kannada & Malayalam) and others.

ANGLABHARTI

- A set of rules obtained through corpus analysis is used to identify plausible constituents with respect to which movement rules for the ‘pseudo-target’ is constructed.
- Within each group the languages exhibit a high degree of structural homogeneity.
- The similarity has been exploited to a great extent in the system.
- A language specific text-generator converts the 'pseudo-target' code into target language text.
- Paninian framework based on Sanskrit grammar using Karak (similar to case) relationship provides an uniform way of designing the Indian language text generators.

ANUBHARTI

- In 1995, Prof. Sinha developed the ANUBHARTI methodology that follows the example based machine translation strategy.
- This methodology has been used for Hindi to English translation.
- The AnuBharti approach works more efficiently for similar languages such as among Indian languages.
- In such cases the word-order remains the same and one need not have pointers to establish correspondences.

ANUBHARTI

- Both of these system architectures, AnglaBharti and AnuBharti, have undergone a considerable change from their initial conceptualization.
- In 2004, phase-II of the system development has been launched which addresses many of the shortcomings of the earlier architectures.
- These are named AnglaBharti-II and AnuBharti-II.
- Both these systems are hybridized with varying degree of hybridization of different paradigms.

ANUBHARTI

- AnglaBharti-II uses a generalized example-base (GEB) for hybridization besides a raw example-base (REB).
- The AnuBharti strategy has now been generalized in AnuBharti-II to cater to Hindi as source language for translation to any other language, though the generalization of the example-base is dependent upon the target language.
- The core of AnuBharti-II architecture is a generalized hierarchical example-base.
- Development of the example-base for Hindi to other Indian languages is lot easier as compared to a dissimilar language.
- Hindi like all other Indian languages is a relatively free word-group order language.
- The input Hindi sentence is converted into a standardized form by shallow grammatical analysis to take care of word-order variations.

ANUBHARTI

- Besides these, they are also currently engaged in development of translation system for bi-lingual text in Hinglish (Hindi mixed with English) and system for speech to speech translation.
- The contact person is Prof. R M K Sinha (rmk@iitk.ac.in).

MANTRA (MAchiNe assisted TRAnslation tool)

- MANTRA (MAchiNe assisted TRAnslation tool) translates English text into Hindi in a specified domain of personal administration, specifically gazette notifications, office orders, office memorandums and circulars.
- Mantra uses Lexicalized Tree Adjoining Grammar (LTAG) to represent the English as well as the Hindi grammar.

MANTRA (MAchiNe assisted TRAnslation tool)

- It uses Tree Adjoining Grammar (TAG) for parsing and generation.
- The input to the Mantra system can be through text documents, outputs of speech recognition programs or OCR packages.
- The Mantra has become part of “The 1999 Innovation Collection” on Information Technology at Smithsonian Institution’s National Museum of American History, Washington DC, USA.
- Further details about the Mantra system can be obtained from the C-DAC website (http://www.cdac.in/html/aai/mantra.asp).

MANTRA (MAchiNe assisted TRAnslation tool)

- The system namely MANTRA-Rajyasabha Is developed for the Rajya Sabha Secretariat, the Upper House of Parliament of India.
- It translates the parliament proceedings such as Papers to be Laid on the Table [PLOT], Bulletin Part-I, Bulletin Part-II, List of Business [LOB] and Synopsis.
- This project was funded by the Rajya Sabha Secretariat.

MANTRA (MAchiNe assisted TRAnslation tool)

- Recently, work has been initiated on other language pairs such as English-Bengali, English-Telugu, English-Gujarati, Hindi-English and among India languages such as Hindi-Bengali, and Hindi-Marathi.
- The contact person is Dr Hemant Darbari (darbari@cdac.in).

ANUSAARAKA MAT System

- A machine aided translation system (ANUSAARAKA) among Indian languages has been built with funding from the TDIL project.
- Anusaaraka is a Language accessor rather than a machine translation system in true sense.
- Anusaarakas have been built from Telugu, Kannada, Tamil, Bengali, Marathi, and Punjabi to Hindi.
- Alpha versions of all of these have been released so that their field testing can be carried out.
- The beta-versions are expected to be released soon.

SHIVA and SHAKTI MT System

- Two machine translation systems from English to Hindi, Shiva And Shakti are being developed jointly by Carnegie Mellon University USA, Indian Institute of Science, Bangalore, India, and International Institute of Information Technology, Hyderabad.
- The two experimental systems have been released for experiments, trials, and user feedback.

SHIVA and SHAKTI MT System

- The Example-based Machine Translation system (Shiva) can be run from the site (http://ebmt.serc.iisc.ernet.in/mt/login.html) and the machine translation system (Shakti) can be run from its own site (http://shakti.iiit.net).
- New Release - 18th April, 2005) is working for three target languages (Hindi, Marathi and Telegu).
- The user has to log on to the system for translating English sentences into the appropriate language.
- The system asks the user to choose a font from a list and shows the translated output in the chosen font.

UNL-based English-Hindi MT System

- The Indian Institute of Technology, Bombay has developed a machine translation system for translating from English to Hindi using the Universal Networking Language (UNL) as the interlingua.
- The UNL is an international project of the United Nations University, with an aim to create an Interlingua for all major human languages.
- IIT Bombay is the Indian participant in UNL.

UNL-based English-Hindi MT System

- The MT system can be used at the site (http://www.cfilt.iitb.ac.in/machine-translation/ eng-hindi-mt).
- Two other demo systems can also be run from the site: one for Hindi-UNL and another for UNL-Hindi conversion.
- They are also working on MT systems from English to Marathi and Bengali using the UNL formalism.
- The contact person is Prof. Pushpak Bhattacharyya (pb@cse.iitb.ac.in)

MATRA MT System

- The Natural Language group of the Knowledge Based Computer Systems (KBCS) division at the National Centre for Software Technology (NCST), Mumbai (currently CDAC, Mumbai) is working on MaTra, human-aided transfer-based translation system for English to Hindi.
- The work is supported under the TDIL Project. The MaTra lexicon and approach is general-purpose, but the system has been applied mainly in the domains of news, annual reports and technical phrases, and has been funded by TDIL.

MATRA MT System

- A Hybrid approach system Vaakya has been developed at NCST, Bombay for translating English news stories to Hindi. The system can handle single verb sentences.
- Prototype Vaakya system is now being enhanced and adapted for providing web translation service to the news agencies from English news stories to Hindi.
- More details about the MaTra system can be obtained from the website of the NCST (http://www.ncst.ernet.in/matra/)
- They are using the translation system in a project on Cross Lingual Information Retrieval (CLIR) that enables a person to query the web for documents related to health issues in Hindi.
- Details about the CLIR system can be seen in (http://www.ncst.ernet.in/projects/clir/). The contact person is Ms. Kavitha Mohanraj (kavitham@ncst.ernet.in).

English-Kannada MT System

- The Computer and Information Sciences Department at the University of Hyderabad has worked on an English-Kannada MT system, using the Universal Clause Structure Grammar (UCSG) formalism.
- This is essentially a transfer-based approach, and has been applied to the domain of government circulars, and funded by the Karnataka government.
- Further details about the English- Kannada machine aided translation system can be obtained from their website (http://www.languagetechnologies.ac.in/lerc/mat/mat.htm).
- Currently, efforts are on to improve the system and apply it for English-Telugu translation as well. The contact person is Prof. K Narayana Murthy (knmcs@uohyd.ernet.in).

Tamil-Hindi MAT System

- The Anna University KB Chandrasekhar Research Centre (http://www.au-kbc.org/) at Chennai has developed a Tamil-Hindi machine aided translation system which can be accessed at (http://www.aukbc.org/research_areas/nlp/demo/mat/).
- There is a keyboard for input of the TamilText and the output can be seen in a Hindi font downloadable from the site.
- The translation system is Anusaaraka based and work on an English-Tamil human aided machine translation system is going on.
- The contact person is Prof. CN Krishnan (cnkrish@au-kbc.org).

ANUVAADAK MT System

- Super Infosoft Private Limited, Delhi have developed a general purpose English-Hindi machine translation tool Anuvaadak 5.0 that supports post-editing.
- It has inbuilt dictionaries for specific domains e.g. official, formal, agriculture, linguistics, technical, and administrative.
- When Hindi meaning of the English word is not available in dictionary, facility of transliteration is provided.
- The software runs on any operating system in the Windows family.
- A demonstration version of the system can be downloaded from the website (http://www.mysmartschool.com/pls/portal/portal .MSSStatic.ProductAnuvaadak).
- The contact person is Mrs. Anjali Rowchowdhury (anjalir@ndf.vsnl.net.in).

Indian Language to Indian Language Machine Translation system (IL-IL MT system)

- The ILMT system is a web-based machine translation system which will produce machine translation (MT) from one Indian language (IL) to another Indian language

Indian Language to Indian Language Machine Translation system (IL-IL MT system

The system will be bi-directional and will work in general domain for the following chosen language pairs.

- Tamil–Hindi
- Telugu – Hindi
- Marathi – Hindi
- Bengali – Hindi
- Tamil – Telugu
- Urdu – Hindi
- Kannada – Hindi
- Punjabi – Hindi
- Malayalam – Tamil

Indian Language to Indian Language Machine Translation system (IL-IL MT system)

- The expected user of the system will be an Indian language speaker. This system will be capable of further improvement using machine learning techniques.
- The system would also deliver MT in two specific domains: (a) tourism and pilgrimage and (b) health. Some basic tools like: POS tagger, chunkers, Morph-analyzer etc would be built as part of the ILMT system.

Indian Language to Indian Language Machine Translation system (IL-IL MT system)

Scope

The system will work on web pages or text material from books, magazines, newspaper etc, written in standard language.

Indian Language to Indian Language Machine Translation system (IL-IL MT system)

Methodology

- The ILMT system is being built by contributors of various NLP research groups.
- The large task of building system is sub-divided into smaller tasks.
- Each participant group takes up one or more task and work in parallel.

Indian Language to Indian Language Machine Translation system (IL-IL MT system)

- Most of the modules are subdivided into a language independent engine and language specific data.
- This allows a software group to concentrate on building the engine.
- Similarly, a language group can contribute data for a specific language.
- The same engine works with different languages.

Indian Language to Indian Language Machine Translation system (IL-IL MT system)

Modularity

- The ILMT system consists of many modules, each one of which typically performs a logical task, usually the task is also small, so that when there is a change it is easier to make it on a small sized module.

Indian Language to Indian Language Machine Translation system (IL-IL MT system)

Dashboard

- All the modules of ILMT run with the help of dashboard, a tool, which configures the data flow pipelines as specified.
- However, it can also speed up the system by setting up shared memory or blackboard, on which all the modules operate, instead of pipelines.
- This however, requires the modules are written to operate on shared memory.

Indian Language to Indian Language Machine Translation system (IL-IL MT system)

System structure

- The Indian language to Indian language machine translation system is based on analyze-transfer-generate paradigm.
- First, analysis of source language would be done, then a transfer of vocabulary and analyzed structure to target language would be carried out, and finally the target language would be generated.

Indian Language to Indian Language Machine Translation system (IL-IL MT system)

- The expected user of the system will be an Indian language speaker. This system will be capable of further improvement using machine learning techniques.
- The system would also deliver MT in two specific domains: (a) tourism and pilgrimage and (b) health. Some basic tools like: POS tagger, chunkers, Morph-analyzer etc would be built as part of the ILMT system.

Scope

The system will work on web pages or text material from books, magazines, newspaper etc, written in standard language.

Methodology

- The ILMT system is being built by contributors of various NLP research groups.
- The large task of building system is sub-divided into smaller tasks.
- Each participant group takes up one or more task and work in parallel.

Methodology

- Most of the modules are subdivided into a language independent engine and language specific data.
- This allows a software group to concentrate on building the engine.
- Similarly, a language group can contribute data for a specific language.
- The same engine works with different languages.

Modularity

- The ILMT system consists of many modules, each one of which typically performs a logical task, usually the task is also small, so that when there is a change it is easier to make it on a small sized module.

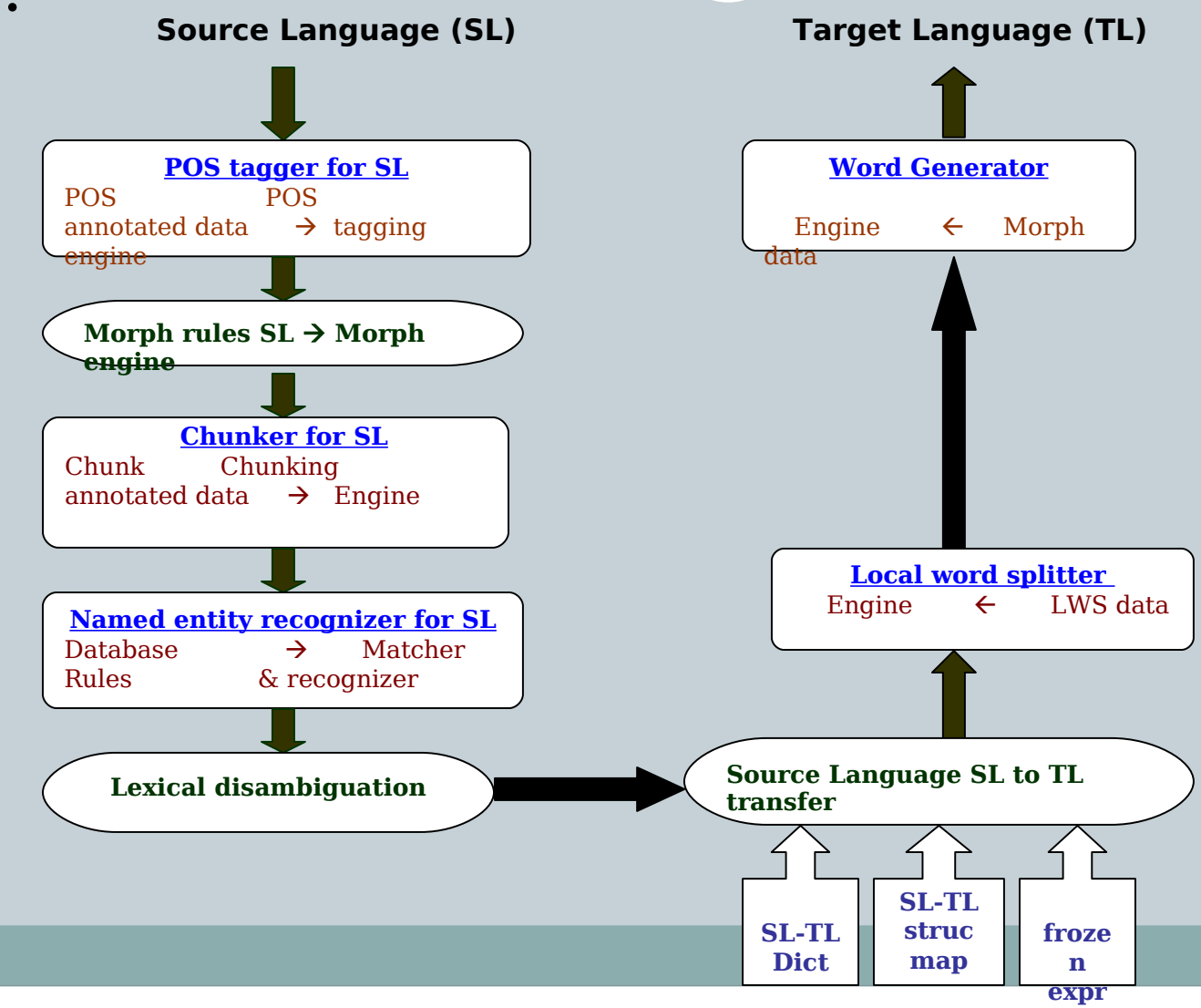
Dashboard

- All the modules of ILMT run with the help of dashboard, a tool, which configures the data flow pipelines as specified.
- However, it can also speed up the system by setting up shared memory or blackboard, on which all the modules operate, instead of pipelines.
- This however, requires the modules are written to operate on shared memory.

System structure

- The proposed Indian language to Indian language machine translation system will be based on analyze-transfer-generate paradigm.
- First, analysis of source language would be done, then a transfer of vocabulary and analyzed structure to target language would be carried out, and finally the target language would be generated.

Architecture


Source Language (SL)
Target Language (TL)
POS tagger for SL
POS annotated data → POS tagging engine
Morph rules SL → Morph engine
Chunker for SL
Chunk annotated data → Chunking Engine
Named entity recognizer for SL
Database Rules → Matcher & recognizer
Lexical disambiguation
Source Language SL to TL transfer
Local word splitter
Engine ← LWS data
Word Generator
Engine ← Morph data
SL-TL Dict
SL-TL struc map
froze n expr

Specification of Individual Modules

- Tokenizer

This module convert a sentence into word level tokens and return sentence marker for each sentence of input text.

Input: Output Specification:

Input: Sentence

Output: Word level Token and Sentence id.

Specification of Individual Modules

Sandhi Splitter

- This module is used for those language which are more agglutinating such as Malayalam, Tamil etc.
- This will take a word as input and its output will be split words.

Input: Output Specification:

Input: Words given by tokenizer is SSF

Output: Split word

Specification of Individual Modules

Morphological Analyzer

- Given a word, the morphological analyzer identifies the root and the grammatical features of the word.
- For languages that are not rich in inflections, a simple lookup dictionary that contains all the word forms would be sufficient.
- For a language like Malayalam the best alternative is to have a dictionary of root words and employ an analyzer to split the word form into its root and grammatical suffixes.
- Paradigm based approaches have given good success rates to Indian languages.

Specification of Individual Modules

Morphological Analyzer

Input-output Specification:

Input: TKN_

Output:af

af is a composite attribute consisting of root, lcat, gend, num, pers, case, Cm / tam, Suffix.

Specification of Individual Modules

Morphological Analyzer

Example

Input

ADDR_	TKN_	CAT_
1	us	<Undef>
2	laTkA	<Undef>
3	ne	<Undef>
4	phaL	<Undef>
5	khAyA	<Undef>

Output

ADDR_	TKN_	CAT_	OTHER
1	us	<Undef>	<fs af=‘ us, pron…’>
2	laTkA	<Undef>	<fs af=‘ laTkA, noun…’>
3	ne	<Undef>	<fs af=‘ ne, preposition…’>
4	phal	<Undef>	<fs af=‘ phal, noun…’>
5	khAyA	<Undef>	<fs af=‘ khAyA, verb…’>

Specification of Individual Modules

POS tagger

- Part of speech tagging is the process of assigning a part of speech to each word in the sentence.
- Identification of the parts of speech as nouns, verbs, adjectives, adverbs for each word of the sentence helps in analyzing the role of each constituent in a sentence.
- There are a number of approaches, such as rule based, transformation-based etc. which are used for POS tagging.
- Here we propose to use statistical approaches.

Specification of Individual Modules

POS tagger

Input-Output Specification:

Input: TKN_ (A few of the morphological features may be included here for different languages.)

Output: CAT_

- The output specifications above means that attribute ‘CAT_’ will get defined by this module.
- (As already explained in section on SSF, **‘af’** is a composite attribute consisting of **root, lcat, gend, num, pers, case**)

Specification of Individual Modules

POS tagger

Example

Input

ADDR_	TKN_	CAT_	OTHER
1	andta	<Undef>	<fs af=‘ andta, pron…’>
2	paiyan	<Undef>	<fs af=‘ paiyan, noun…’>
3	pazham	<Undef>	<fs af=‘ pazham, noun…’>
4	tinRaan	<Undef>	<fs af=‘ tinkiRaan, verb…’>

Output

ADDR_	TKN_	CAT_	OTHER
1	andta	PRP	<fs af=‘ andta, pron…’>
2	paiyan	NN	<fs af=‘ paiyan, noun…’>
3	pazham	NN	<fs af=‘ pazham, noun…’>
4	tinRaan	VM	<fs af=‘ tinkiRaan, verb…’>

Specification of Individual Modules

Chunker:Chunking:

- Chunking involves identifying simple noun phrases, verb groups, adjectival phrase, and adverbial phrase in a sentence.
- This involves identifying the boundary of chunks and the label.

Specification of Individual Modules

Chunker: Chunking:

Input-Output Specification:

Input: TKN_, CAT_

Output: ADDR_ (This means restructuring takes place, because this is when value of ADDR_ changes)

- The input specifications above means that it uses or requires property TKN_ and CAT_ to be defined in the SSF.
- In other words, the words or tokens in the input sentence together with part of speech category are given.
- The Output specifications above mean that attribute ‘ADDR_’ will be get defined by this module.

Specification of Individual Modules

Chunker: Chunking:

Input

ADDR_	TKN_	CAT_	OTHER
1	andta	PRP	<fs af=‘ andta, pron…’>
2	paiyan	NN	<fs af=‘ paiyan, noun…’>
3	pazham	NN	<fs af=‘ pazham, noun…’>
4	tinRaan	VM	<as af=‘ tinkiRaan, verb…’>

Specification of Individual Modules

Chunker: Chunking:

Output

ADDR_	TKN_	CAT_	OTHER
1	[[	NP	
1.1	andtap	PRP	<fs af=‘ andta, pron…’>
1.2	paiyan	NN	<fs af=‘ paiyan, noun…’>
	]]		
2	[[	NP	
2.1	pazham	NN	<fs af=‘ pazham, noun…’>
	]]		
3	[[	VG	
3.1	tinRaan	VM	<as af=‘ tinRaan, verb…’>
	]]		
	.	SYM	

Specification of Individual Modules

Chunker: Head computation

This module computes the head of the chunk.

Input-Output specification:

Input: TKN_, CAT

Output: lcat [CAT_] -> {head, name}

pcat[CAT_] -> {head}

Specification of Individual Modules

Chunker: Head computation

Example

Input

ADDR_	TKN_	CAT_	OTHER
1	[[	NP	
1.1	andtap	PRP	<fs af=‘ andta, pron…’>
1.2	paiyan	NN	<fs af=‘ paiyan, noun…’>
	]]		
2	[[	NP	
2.1	pazham	NN	<fs af=‘ pazham, noun…’>
	]]		
3	[[	VG	
3.1	tinRaan	VM	<as af=‘ tinkiRaan, verb…’>
	]]		
	.	SYM	

Specification of Individual Modules

Chunker: Head computation:

Output

ADDR_	TKN_	CAT_	OTHER
1	[[	NP	<fs head=2>
1.1	andtap	PRP	<fs af=‘ andta, pron…’>
1.2	paiyan	NN	<fs af=‘ paiyan, noun…’>
	]]		
2	[[	NP	<fs head=1>
2.1	pazham	NN	<fs af=‘ pazham, noun…’>
	]]		
3	[[	VG	<fs head=1>
3.1	tinRaan	VM	<as af=‘ tinkiRaan, verb…’>
	]]		
	.	SYM	

Specification of Individual Modules

Chunker: Inherit head features

All the features are copied from head-daughter to parent chunk except ‘name’.

Input-output specification:

Input: ANY_

Output: ANY_,copied_

Specification of Individual Modules

Chunker: Inherit head features

Example:

Input

ADDR_	TKN_	CAT_	OTHER
1	[[	NP	<fs head=2>
1.1	andtap	PRP	<fs af=‘ andta, pron…’>
1.2	paiyan	NN	<fs af=‘ paiyan, noun…’>
	]]		
2	[[	NP	<fs head=1>
2.1	pazham	NN	<fs af=‘ pazham, noun…’>
	]]		
3	[[	VG	<fs head=1>
3.1	tinRaan	VM	<as af=‘ tinkiRaan, verb…’>
	]]		
	.	SYM	

Specification of Individual Modules

Chunker: Inherit head features

Output

```
ADDR_ TKN_       CAT_       OTHER
  1            [[                NP              <fs af=‘ paiyan, noun…’
 head=paiyan>
 1.1      andtap     PRP        <fs af=‘ andta, pron…’>
 1.2        paiyan      NN          <fs af=‘ paiyan, noun…’>
            ]]
 2                  [[              NP          <fs af=‘ pazham, noun…’
 head=pazham>
 2.1        pazham     NN         <fs af=‘ pazham, noun…’>
            ]]
 3        [[            VG         <fs af=‘ tin, verb…’ head=tinRaan>
 3.1        tinRaan    VM          <as af=‘ tinkiRaan, verb…’>
            ]]
             .              SYM
```

Specification of Individual Modules

Local word Grouper

- Local word grouper does the technical task of vibhakti computation.
- It will group case/tam marker with root for noun and verbs.

  Input-Output specification:

  Input: Output of chunk

  Output: Grouped chunked as noun grouped with its case and verb grouped with its tam.

Specification of Individual Modules

Named entity recognizer

This module identifies the entities and tags them with relevant NER tags. It can process inputs in Hindi, English and Other Indian languages.

Input-output Specification

Input: Pre-processed text

Assignment of appropriate NER tags to each entity in the input text.

Specification of Individual Modules

Simple Parser

This module identifies relation between some of the chunks of the sentence.

Some of these relations will depend on the language being analysed.

Specification of Individual Modules

Lexical Sense Disambiguation

- This module disambiguates each word in the input text and returns the sense tagged text as output.
- It can process input in Hindi, English, or any other Indian language.

Input-out specification:

Input: Input text

Output: the sense tagged text

Specification of Individual Modules

Transfer Module

- The syntactic difference between the language can be found in complex sentence construction particularly participles.
- This would be bridged by means of transfer grammar.
- This component will also include the task of transliteration.

Input-Output specification:

Input: Source Language chunks with their feature structure

Output: Mapped structure of target language as chunk and their feature structure

Specification of Individual Modules

Bilingual mapping

- The root words identified by the morphological analyzer are looked upon in a bilingual dictionary for the target language as well as its category (say noun, verb, etc.), paradigm and other necessary information.
- This stage also uses dictionaries to identify the target language equivalence for the source language grammatical suffixes.

Specification of Individual Modules

Sentence Level Generation

- This task would primarily be reordering the chunks, if required.
- Certain sentence level aspects will also be handled by this module.
- For example, noun-verb agreement etc will be handled here.
- This phase is above chunk level processing.

Specification of Individual Modules

Morphological generation

- In order to generate a word form for a specific grammatical category the corresponding suffixes have to be concatenated with the root word.
- The morphological generator then inflects the root word according to the morphology of the language and ouptputs the target language word form.
- The word thus generated are concatenated to form the complete target language sentence.

Specification of Individual Modules

Evaluation

- Evaluation mechanism would be setup for end users for individual sentences as well as text.
- This should allow subjective evaluation, for comprehensibility and quality of the translation.
- This second stage, evaluation would be done on speed and usability of the system.

Language Coordinators (Verticals)

- IITB Marathi
- IIITKGP Bengali
- HCU Telugu
- AU-KBC Tamil
- IIITH Hindi
- IISc Kannada
- RSVP Urdu
- Tamil U Malayalam

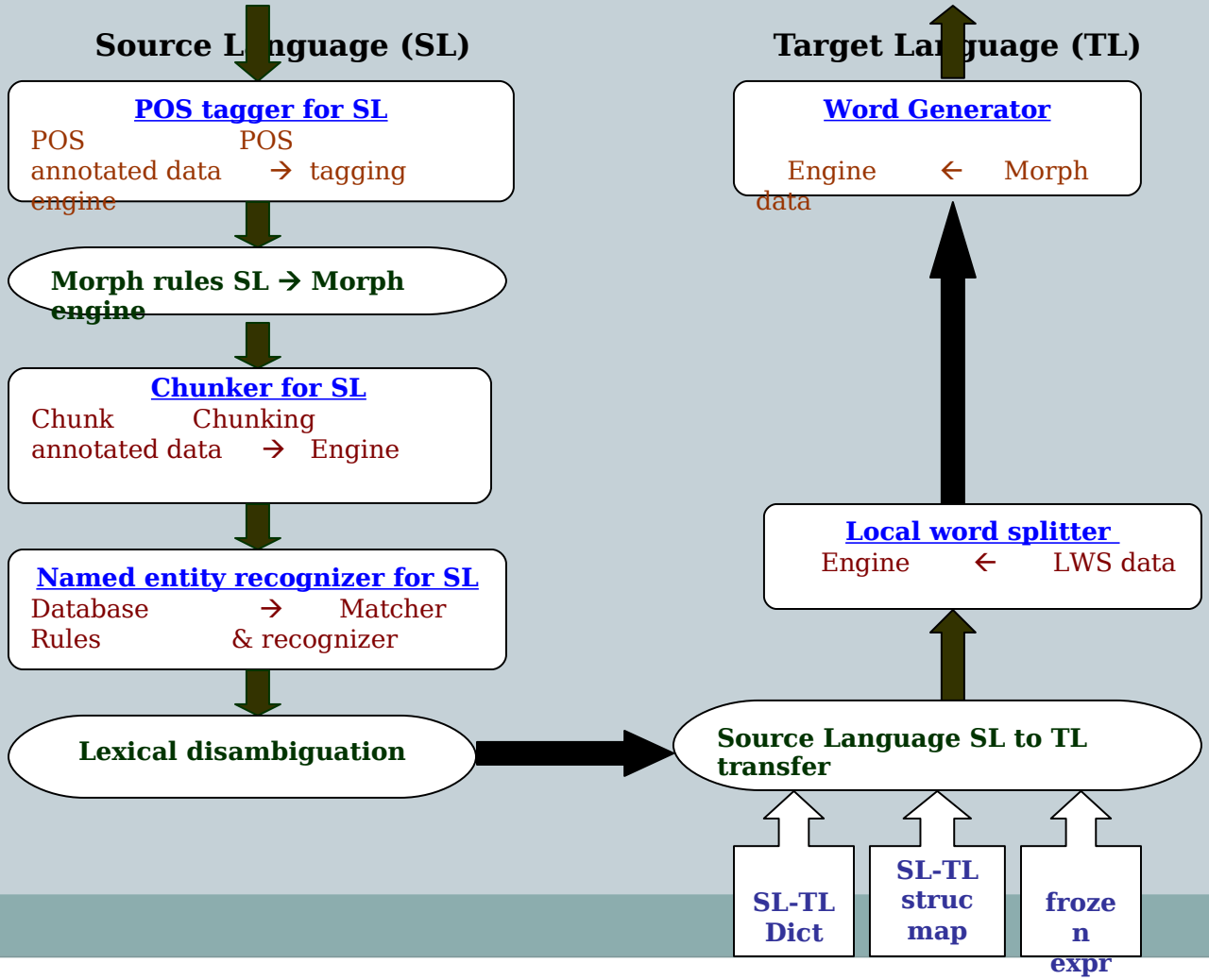
Language Verticals

- Marathi
  - IITB*, CDACP (* => coordinator)
- Bengali
  - IITKGP*, CDACP, JU, CDACK?
- Telugu
  - HCU*, IIITH
- Tamil
  - AU-KBC*, AU-CEG
- Hindi
  - IIITH*, CDACP

Language Verticals

- Kannada
  - IISc*, Tamil U(?)
- Urdu
  - RSVP Tirupati*, CDACP, IIITA
- Malayalam
  - Tamil U*

Architecture


Source Language (SL)
POS tagger for SL
POS annotated data → POS tagging engine
Morph rules SL → Morph engine
Chunker for SL
Chunk annotated data → Chunking Engine
Named entity recognizer for SL
Database Rules → Matcher & recognizer
Lexical disambiguation
Target Language (TL)
Word Generator
Engine ← Morph data
Local word splitter
Engine ← LWS data
Source Language SL to TL transfer
SL-TL Dict
SL-TL struc map
froze n expr

Horizontals

- POS Tagger engine
- Chunking engine
- Morph analyser engine
- Lexical disambiguation engine
- Named entity engine
- Dictionary standards
- Annotated corpora standards
- Evaluation of output (comprehensibilty)
- Testing & integration

Horizontal Tasks

- IITB
  - Lexical disambiguation engine
  - Dictionary standards
- IITkgp
  - Word generator engine
  - Annotated corpus standards
- HCU
  - Morph Analyser engine
- AUKBC
  - Named Entity engine
- IIITH
  - POS tagger engine
  - Chunker engine
  - Testing and system integration
- CDAC P
  - Evaluation output - standards

IL to IL Pairs

- Marathi - Hindi (Bidirectional)
- Bengali - Hindi (,,)
- Telugu - Hindi (,,)
- Tamil- Hindi (,,)
- Tamil- Telugu (,,)
- Kannada – Hindi ('')
- Urdu - Hindi ('')
- Malayalam – Hindi ('')

Vertical Tasks for Each Language

- POS tagger
- Chunker
- Morph analyzer
- Generator
- Named entity filter
- Bilingual dictionary – both directions
- Transfer grammar rules
- Annotated corpus
- Evaluation of output

Discussion!!!